Microsoft Windows

HORUS: Large Scale SMP Using AMD® Opteron™ Processors

Rajesh Kota
Newisys Inc., a Sanmina-SCI Company

April 24, 2005

Abstract

This paper describes the Newisys implementation of an ASIC called HORUS. The HORUS chip enables building large-scale, partitionable, distributed, symmetric multiprocessor (SMP) systems using AMD® Opteron™ processors as building blocks. HORUS scales the glueless SMP capabilities of Opteron processors from 8 sockets to 32 sockets. HORUS provides high-throughput, coherent HyperTransport™ (cHT) protocol handling using multiple protocol engines (PE) and a pipelined design. HORUS implements reliable fault tolerant interconnect between systems using InfiniBand (IB) cables running Newisys extended cHT protocol. HORUS implements a directory structure to filter unnecessary Opteron cache memory snoops (probes) and 64MB of remote data cache (RDC) to significantly reduce the overall traffic in the system and reduce the latency for a cHT transaction. A detailed description of the HORUS architecture and features are discussed followed by results of performance simulations.

Contents

Windows Hardware Engineering Conference

Introduction

The AMD® Opteron™ processor has integrated on-die memory controller, Host Bridge and three HyperTransport™ (HT) links. The three HT links provide glueless SMP capability to eight nodes. A node for this article represents a physical socket. The memory controller is dual ported and supports DDR-SDRAM physical memory. The host bridge provides the interface between the coherent domain (between the processors) and the non-coherent domains (I/O).

In a SMP system using MP-enabled Opteron processors, the physical memory is spread across different memory controllers. Each memory controller is home to a range of physical addresses. Each Opteron could potentially have an IO chain connected to its host bridge. Each processor has address mapping tables and routing tables. The address mapping table maps nodes to physical memory or IO region. Routing tables map HT links to nodes for routing HT packets.

Opteron processors are limited by cHT protocol to support only eight nodes. Opteron processors have very good scaling to at least four way SMP systems. Performance of important commercial applications is challenging above 4-way due to link interconnect topology (wiring and packaging) and link loading with less than full interconnection. Going above 8-way SMP requires fixing both the address-ability of nodes beyond eight in cHT and better interconnect topology. Newisys solves this in the implementation of HORUS.

HORUS ExtendiScale Architecture

HORUS cache coherence (CC) protocol merges multiple 4-way Opteron systems into a larger, low latency CC system. HORUS CC protocol is a new protocol built on top of cHT protocol that will enable merging of multiple cHT-based quads into a single CC system. HORUS CC protocol provides functions beyond the cHT protocol. These functions include remote data caching, CC directory, optimized quad-to-quad protocols, and a quad-to-quad guaranteed exactly once packet delivery mechanism.

Figure 1: Interconnections between HORUS and Opteron processors inside a quad

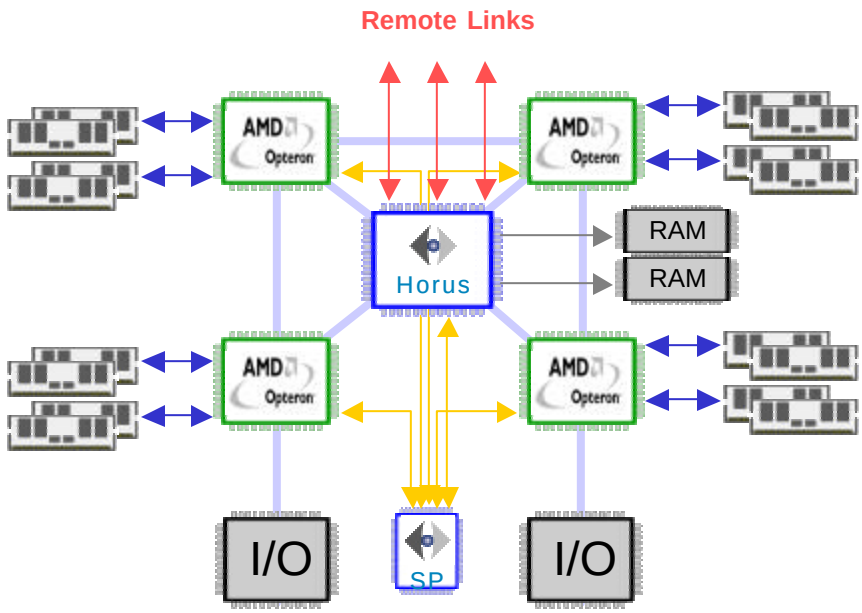


Figure 1 shows the interconnections between HORUS and Opteron inside a quad. The HORUS chip has four cHT links and three remote links. Each cHT is a 16bit lane source synchronous interface running at 2GHz. The remote links provide interconnection between the quads. Each remote link is implemented using 12 lanes of 3.125GHz serializers and deserializers (SerDes) physicals with 8b/10b encoding providing an effective data rate of 30 Gbps. Standard InfiniBand cables are used for interconnection. The HORUS CC protocol is limited up to eight quads (32 sockets total). Figure 2 shows the different configurations that are possible using HORUS. Figure 3 shows one such implementation in Blade based system.

Figure 2: Different configurations with HORUS

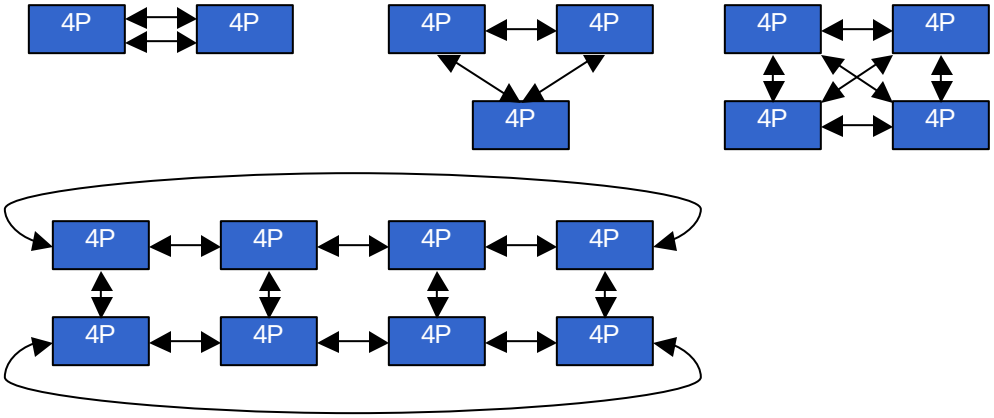
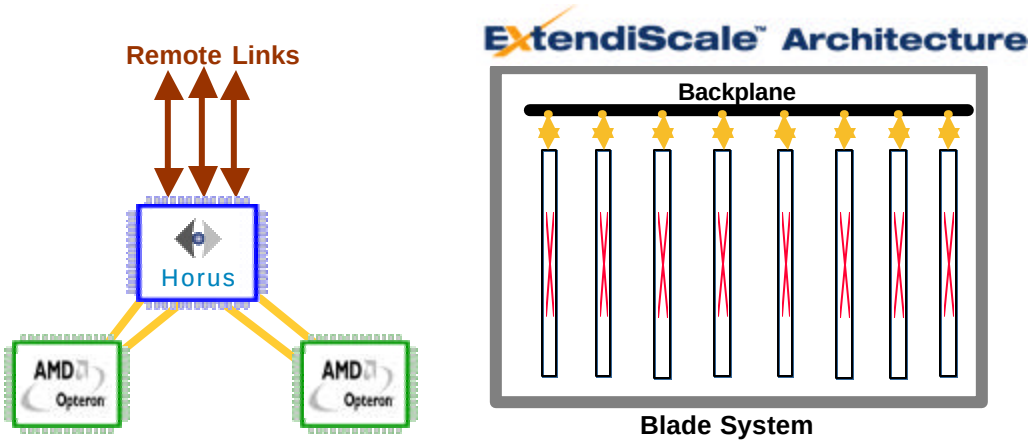


Figure 3: HORUS extending Opteron to larger SMP blade systems



Address Mapping and Packet Retagging

HORUS looks like another Opteron processor to the local Opteron processors in a quad. HORUS acts as a proxy for all the remote memory controllers, CPU cores and Host Bridges. The mapping tables in the local Opteron processors are programmed by Newisys BIOS to direct requests to physical memory or IO residing in remote quads to HORUS. Opteron processors in the local quad are not aware of the remote Opteron processors.

In a quad using HORUS, each active memory controller is assigned a contiguous range of physical addresses and each quad is assigned one contiguous range of physical addresses. The physical address region above and below this quad's region is assigned to HORUS in local Opteron's address mapping tables. The global address mapping tables in HORUS contains information of which Quad is assigned what regions of physical address.

Each transaction in cHT domain consists of multiple packets from different Opteron processors. The packet type could be requests, probes, broadcasts or responses. Some packets have data associated with them and some don't. All packets that belong to one transaction have a unique and common transaction ID. This is essential so that Opteron processors can stitch together all the packets related to a particular transaction.

Transactions generated by an Opteron in one quad could have the same transaction ID as a completely different transaction generated by another Opteron in a different quad. When a transaction goes through HORUS from the local domain to remote domain (or vice versa) a new transaction ID is created and substituted for the incoming transaction ID. Each HORUS maintains mapping between the two different transaction IDs between the two domains (local and remote).

Transaction Flow Examples

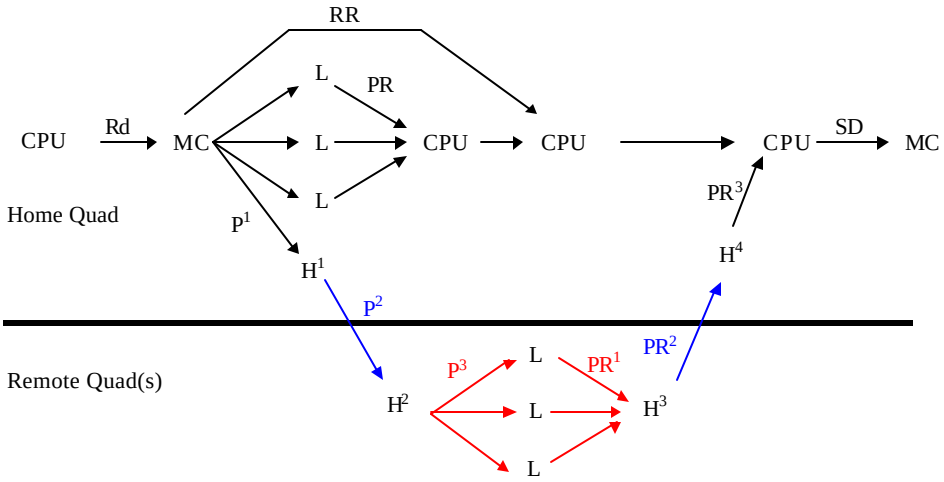
Remote Probing

Figure 4 shows how HORUS extends the cHT protocol for a request from local CPU to local memory controller with remote probing. The notation used for the protocol diagrams used in this document is as follows:

L	A local Opteron processor (which includes the CPU and its caches, MC, HB)
CPU, MC	The CPU and Memory Controller inside Opteron processors
H	HORUS Chip
Rd	Read Request
RR	Read Response (has data)
PR	Probe Response (has no data only status info.)
P	Probe (Opteron cache memory snoop. Response to probe is sent to the source CPU or target MC depending on the request type. The response may either be a probe response or read response with data if the line was dirty in the Opteron cache)
SD	Source Done (Transaction committed at source)

HORUS at Home quad (H^1) receives probe (P^1) for a local memory request and forwards probe (P^2) to all remote quads. HORUS at remote quad(s) (H^2) receives remote probe (P^2) and broadcasts probe (P^3) to all local nodes. HORUS at remote quad(s) (H^3) accumulates CPU probe responses (PR^1) and forwards accumulated response (PR^2) back to home node. HORUS at home quad (H^4) accumulates responses (PR^2) from remote quad(s) and forwards accumulated response (PR^3) back to requesting CPU.

Figure 4: Local Processor to Local Memory (LPLM) example



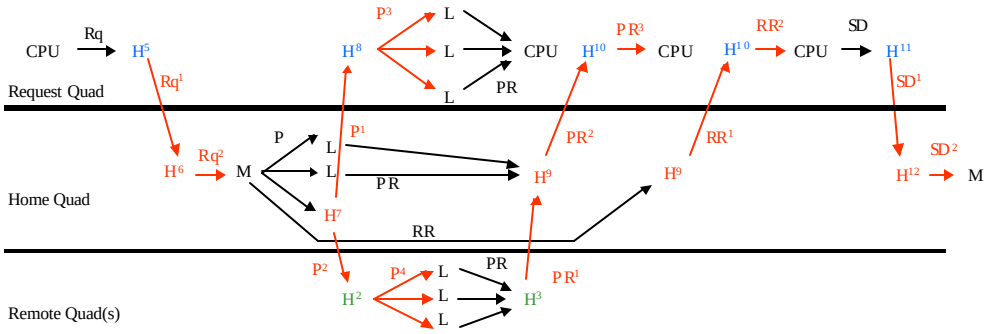
H^1 and H^4 are the same physical HORUS in Home Quad at different stages of the transaction. Similarly H^2 and H^3 are the same physical HORUS in Remote Quads at different stages of the transaction.

Remote Fetching

A transaction from local processor to remote memory is shown in figure 5. HORUS at local quad (H^5) accepts local processor (CPU) request (Rq) that targets remote memory and forwards request (Rq^1) to home quad HORUS (H^6). H^6 receives Rq^1 and forwards request (Rq^2) to home memory controller. Memory controller generates and forwards probe (P) to local nodes, include HORUS (H^7). H^7 forwards it to remote quads, requesting quad (P^1) and other remote quads (P^2).

HORUS at requesting quad (H^8) accepts probe (P^1) and forwards to local nodes. HORUS at remote quads (H^2) accepts probe (P^2) and forwards probe (P^4) to local nodes. HORUS at remote quad (H^3) accumulates responses and forwards accumulated response (PR^1) to home quad. HORUS at home quad accumulates responses from local nodes and remote quads (PR^1). Once all probe responses have been received, an accumulated probe response (PR^2) is forwarded to requesting quad. HORUS also forwards the memory controller response to requesting quad (RR^1).

Figure 5: Local processor request to remote memory (LPRM) example



HORUS (H^{10}) at requesting quad accepts accumulated probe response (PR^2) and forwards response (PR^3) to requesting node (CPU). HORUS (H^{10}) also accepts read response (RR^1) from memory controller and forwards RR^2 to requesting node (CPU).

After requesting node receives responses from all local nodes and a read response from the memory controller, it completes the transaction by generating a source done (SD) response back to memory controller, HORUS (H^{11}) in this example. H^{11} forwards response (SD^1) to home quad HORUS (H^{12}). H^{12} forwards SD^2 to memory controller. The transaction is now complete.

Protocols described above are implemented using protocol engines (PE). There are three flavors of PEs. They are local memory PE (LMPE), remote memory PE (RMPE) and special function PE (SPE). The LMPE handles all transactions directed to local memory controllers and local host bridges. The RMPE handles all transactions directed to remote memory controllers and remote host bridges. The SPE has access to internal control register bus and it processes PCI-Configurations.

Performance Enhancements

Features of HORUS described above are essential to extend SMP capabilities of Opteron from 8 to 32 nodes. For performance of a system using HORUS to scale, HORUS needs to address the bandwidth and latency issues in a large SMP system.

Directory

HORUS implements a CC directory inside LMPE. The directory maintains invalid, shared, owned and modified states for each local memory line that is cached remotely. It also maintains an occupancy vector, with one bit per quad tracking which quad has a cached copy of the memory line. Probes will be sent only to quads where memory line may be cached. This function helps to reduce probe bandwidth and transaction latency.

HORUS implements on-die a sparse, two memory lines per entry (2 sectors), and 8-way set-associative directory using repairable SRAM. Sectoring is used to limit the physical size of on-chip tag arrays. Sparcity is sized at 50% for a four quad system. Sparcity is measured as the ratio of total number of memory lines in directory to the size of the caches in remote quads including the remote data caches of remote HORUS chips. The 50% sparcity was a result of trade off between die size and optimal performance in a 4 quad implementation.

Entries are allocated as requests from remote nodes are received, and the entries may be de-allocated if the line is no longer remotely cached or if the entry must be evicted (least recently used entry is evicted) to accommodate a new request. Directory issues a sized write request to evicted memory line to force invalidation or flushing of dirty data from remote caches back to memory.

Figure 6: A local processor to local memory controller completion using DIRECTORY

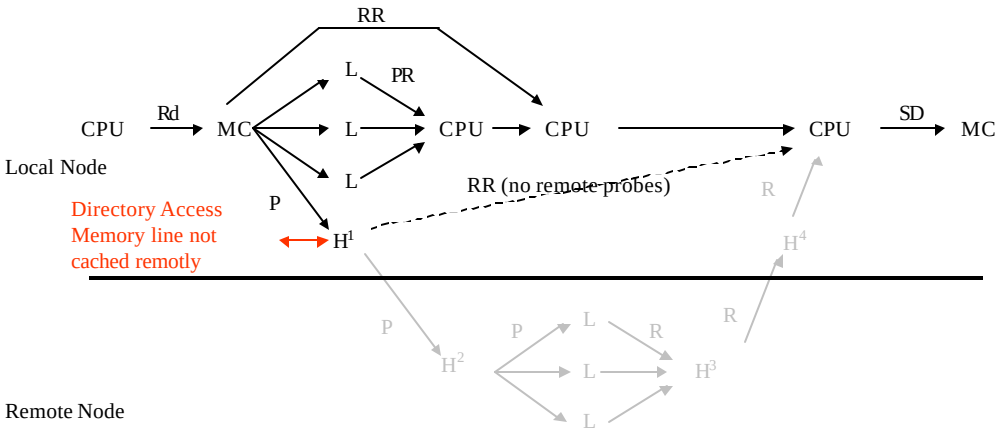
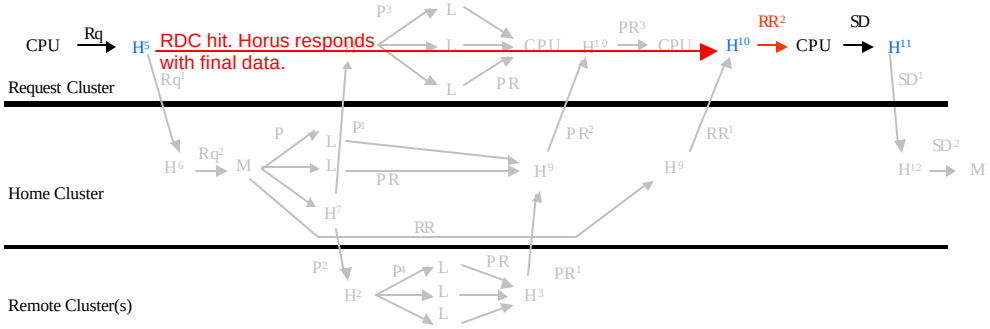


Figure 6 shows a request of a memory line from a local processor to local memory controller that is not cached remotely. In this case, HORUS looks up the memory line in the directory and finds it not cached in any remote quad. Therefore, no remote probes are broadcasted and the transaction is completed quickly in the local quad itself. All the packets eliminated in this transaction due to directory are in gray color. This figure shows the performance advantages of the directory.

Remote Data Cache

HORUS supports 64MB of off-chip RDC. The RMPE have tags on-chip to track data cached in off-chip memory. Tag array has two memory lines per entry (2 sectors), 8-way set associative and implemented using on-chip SRAM and is accessed in one cycle. Off-chip memory was limited to 64MB to keep the directory sparcity at 50% for four quad systems. The RMPE maintains shared, exclusive and invalid states for memory lines held in RDC.

Figure 7: A hit in RDC for remote data request causes the transaction to complete locally



RDC is filled in two ways. For remote requests that miss in RDC, remote protocol keeps a copy of the data returned from memory controller. Also, when a local Opteron processor evicts a dirty memory line from its cache, RMPE keeps a copy of it. RDC implements least recently used policy to evict memory lines during conflict.

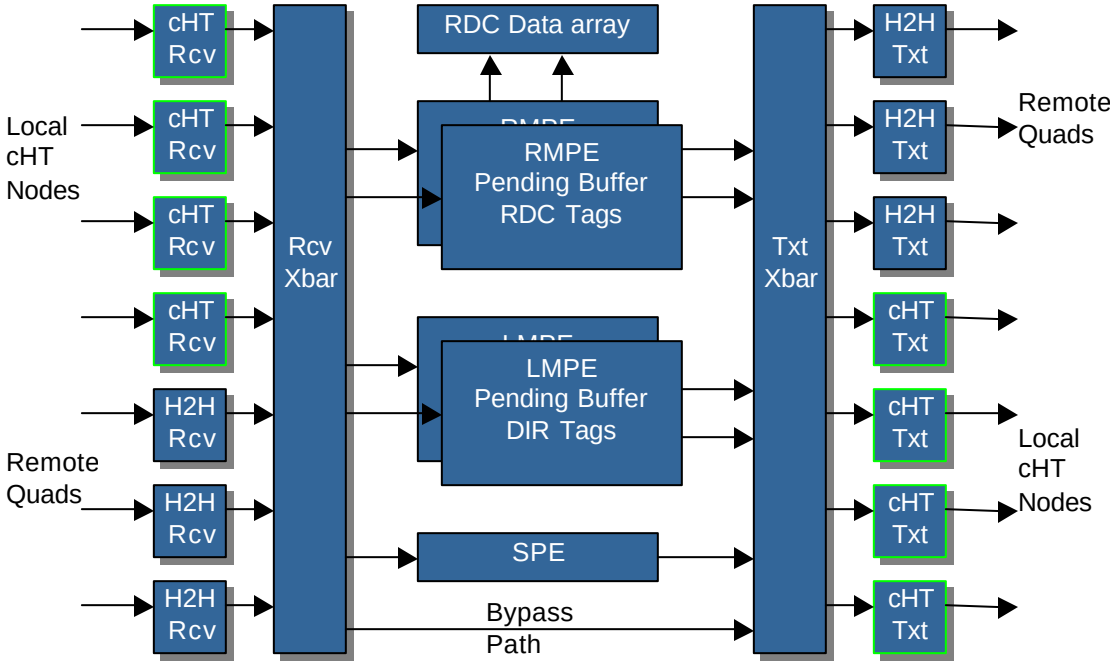
Requests for remote physical data are forwarded to RMPE in HORUS. For shared requests, RMPE checks tag array to see if data is present in RDC. If data is present in RDC, it is forwarded to requesting Opteron and transaction is completed locally as shown in figure 7. The grayed out packets in the above figure show what are eliminated by a hit in RDC. It shows the enormous performance improvements achieved due to RDC. Requests for write permission of memory line are forwarded to remote memory controller so that invalidated probes can be generated by the memory controller to all caches in system having a copy of the memory line.

Microarchitecture

The major functional blocks inside HORUS are the four cHT receiver (cHT Rcv) and transmitter (cHT Txt) links, three remote receiver (H2H Rcv) and transmitter (H2H Txt) links, receiver crossbar (Rcv Xbar), transmitter crossbar (Txt Xbar), RDC data array with dual ported interface, two RMPE engines, two LMPE engines and one SPE.

The cHT transactions consist of command and data packets, which take different paths through HORUS. Figure 8 shows the path taken by command packets through HORUS. The various PE inside HORUS process the command packets only. The data flows directly from the receiver links to the transmitter links. Each data packet is assigned a data pointer inside HORUS. The PE controls the flow of data inside HORUS using the data pointers.

Figure 8: cHT command flow through HORUS



cHT Receiver and Transmitter Links

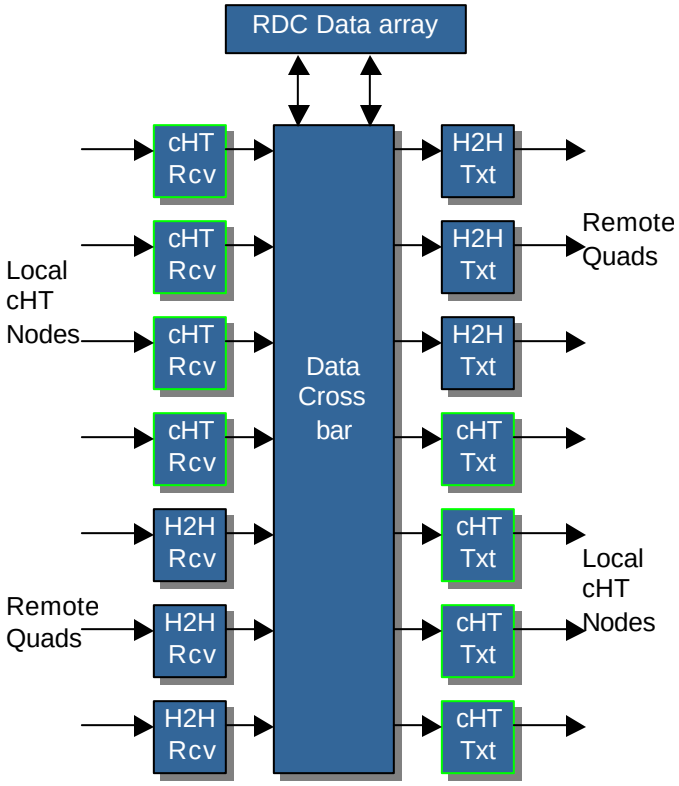
The cHT receivers and transmitters in HORUS comply with the cHT protocols. Receiver modules decode incoming command and data packets. Data packets remain inside the data buffer until transmitter links or the RDC request them. Receiver modules forward command packets to the PE for processing. The cHT receiver and transmitter modules maintain complete separation between packets of different virtual channels specified in cHT.

Remote Receiver and Transmitter Links

As described earlier, HORUS implements three remote links, and only three, to limit I/O pin count and still provide full connectivity for up to four quads.

We have extended the cHT protocol for remote links and separated the remote protocol layer from the remote link layer. The remote link layer is extremely reliable and implements a guaranteed-exactly-once delivery system. Inside HORUS is the hardware support to enable hot plug and hot unplug of the remote links. The software must guarantee that no active transactions are in the system during a hot-unplug event.

Figure 9: cHT data flow through HORUS



Pipelined Protocol Engines

HORUS processes cHT packets through PEs, which let HORUS process transactions simultaneously and thus provides more bandwidth. Each PE is pipelined and has a 32-entry-deep pending buffer, with one entry per transaction. Each transaction can comprise multiple cHT packets, so the pending buffer can keep track of the transaction state and the accumulating responses. Each PE can handle 32 transactions simultaneously.

HORUS implements the cHT protocols in microcode, with the base protocol implemented in a ROM structure. A RAM structure also exists in the PE so that the BIOS can load a completely new protocol. As we described earlier, the BIOS programs a PE to be either a LMPE, RMPE, or SPE—which is actually the same physical PE instantiated multiple times. Each HORUS has two RMPEs, two LMPEs, and one SPE.

Crossbars

As Figures 8 and 9 shows, HORUS has three crossbars, all of which are non-blocking. The receiver crossbar moves command packets from seven receiver links to five PEs. The transmitter crossbar moves command packets from five PEs to seven transmitter links. Data packets don't go through a PE. When they come into HORUS, it buffers them in the receiver link. When a PE processes the command packet associated with a data packet, it either discards the data or forwards it through a transmitter link.

After processing a command packet, the PE sends it to a transmitter link. The transmitter link requests data associated with the command packet from the data

crossbar. The data crossbar moves data from the receiver links directly to the transmitter crossbar.

Bypass Path

The bypass path is used to forward packets directly from receiver links to transmitter links without any modification. This is a very low-latency cut-through path. Bypass paths are completely non-blocking. When Opteron processors in a local quad do not have a direct cHT link between them, the local bypass path in HORUS serves as that link. Because we limited HORUS to three remote links, in configurations with more than four quads, remote bypass links route packets between quads that are not directly connected.

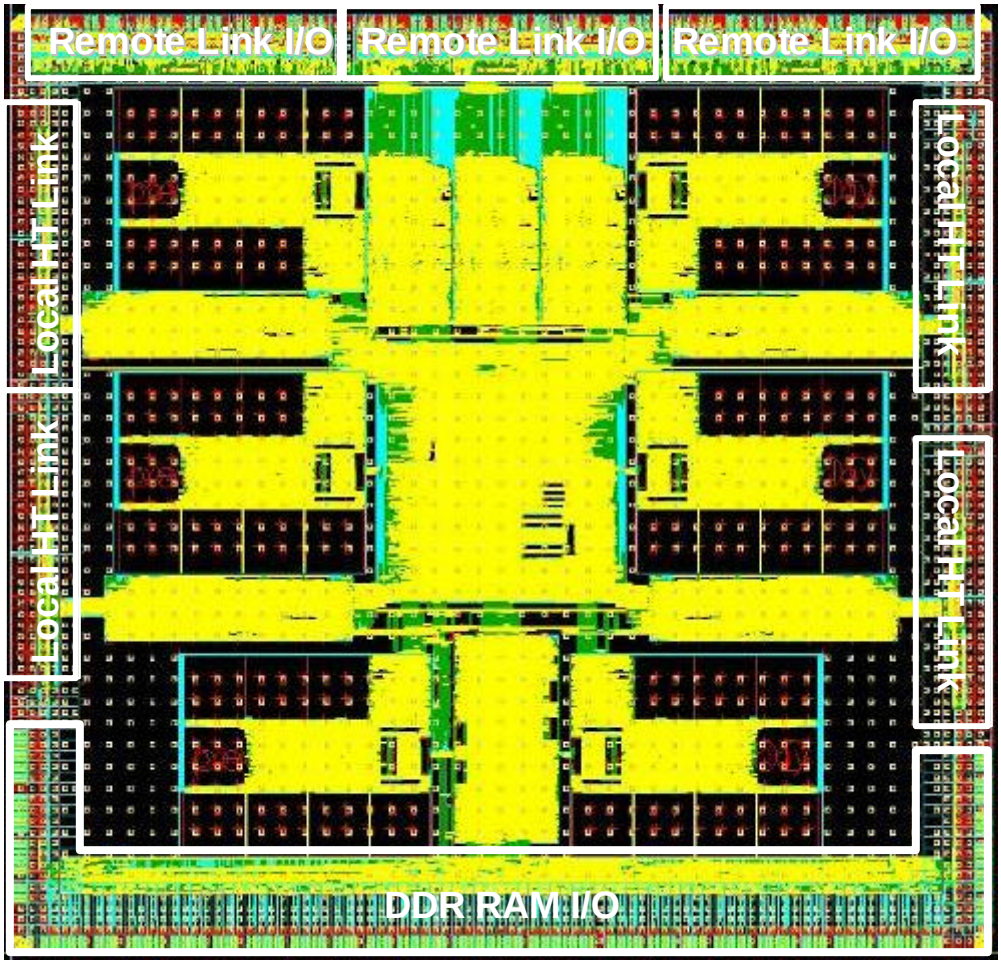
HORUS Vital Statistics and Current Status

Figure 10 shows the A.0 Die plot of HORUS. Total number of signal IO pin count is 730. The power and ground pin count is 479.

HORUS is implemented in 130nm, TSMC, LVOD technology. The core frequency of HORUS is 500MHz. Expected power consumption is ~35Watts. Gate count excluding memory is ~ 10 Million. Transistor count excluding memory is ~ 38 Million. On-chip repairable SRAM size is 3.75 MB (from Virage). The other hard macros are HT, SerDes and PLLs (from ARM).

HORUS RTL was implemented using Verilog. Total number of lines of code is ~ 115K + 50K (auto generated) + 20K (Verilog libraries). Verification infrastructure of HORUS was implemented in Vera, Java and C++. Verification LOC: > 700K (Vera, Java, C++).

Figure 10: HORUS A.0 Die Plot



A.0 silicon samples of HORUS are currently being validated in our Labs. B.0 version of HORUS is expected to tape out in fall of 2005.

Figure 11: HORUS A.0 chips

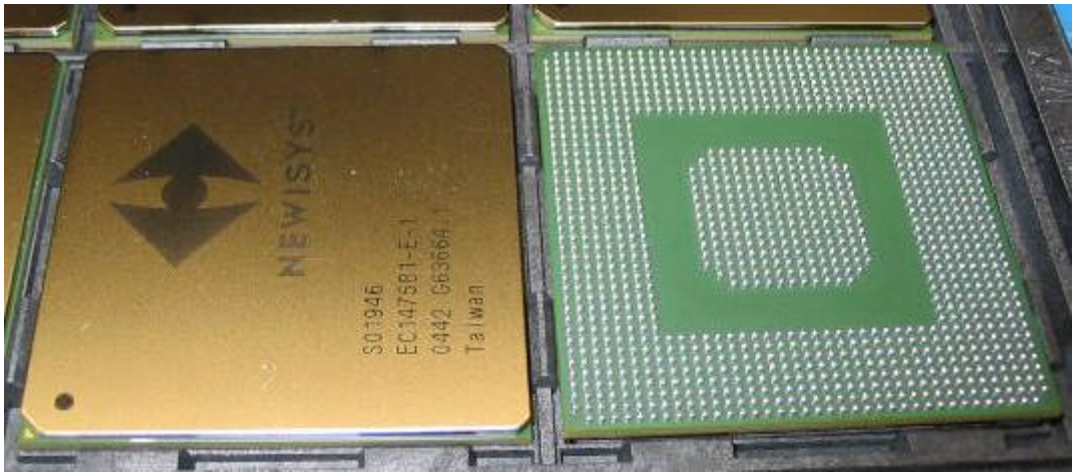
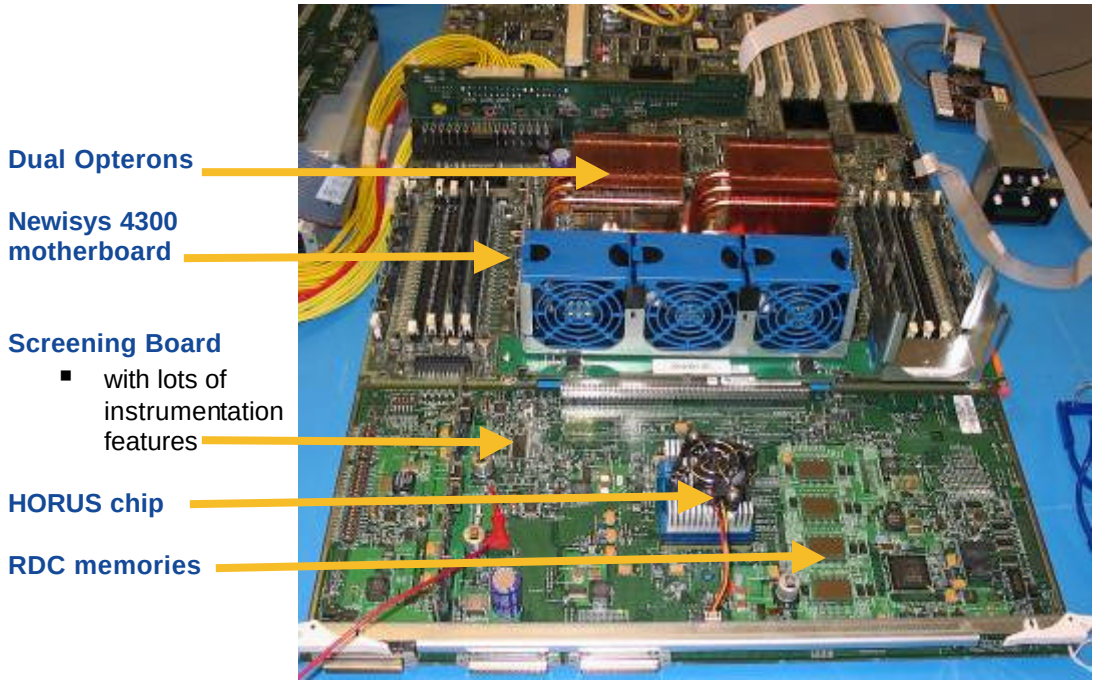


Figure 12: HORUS A.0 bring up board



Features in HORUS for an Enterprise Server

Reliability, Availability, and Serviceability

HORUS is ideally suited for an enterprise SMP system. On the remote links, HORUS can recover from all soft errors without any help, including those caused by disparity; out-of-band signals; loss of signal; first-in, first-out overflows on the physical layer; cyclic redundancy check mismatches; packet loss; packet sequence ID mismatches; and illegal packets. The remote links will also recover from all soft errors.

All arrays (on and off chip) have error checking and correction that supports single-bit error correction, double-bit error detection, and scrubbing. Single-bit errors on a tag-array read will dynamically stretch the PE pipeline to allow for single-bit error correction until the pipeline is idle, when idle the pipeline will shrink to a regular flow.

Machine Check Features

Implemented inside HORUS are extensive error detection, correction and logging features. All errors (both Fatal and Non-Fatal) can be programmed to cause either no action or interrupt SP or flood the links (i.e. Bring the system down).

HORUS has a JTAG (IEEE Std. 1149.1-1990) mailbox interface, which the Newisys service processor can use to periodically monitor HORUS's health. The service processor can disable functionality in HORUS dynamically so that the system can make forward progress at a reduced performance. The JTAG mailbox provides sideband access to all configuration, performance, and debug registers in HORUS.

Performance Counter

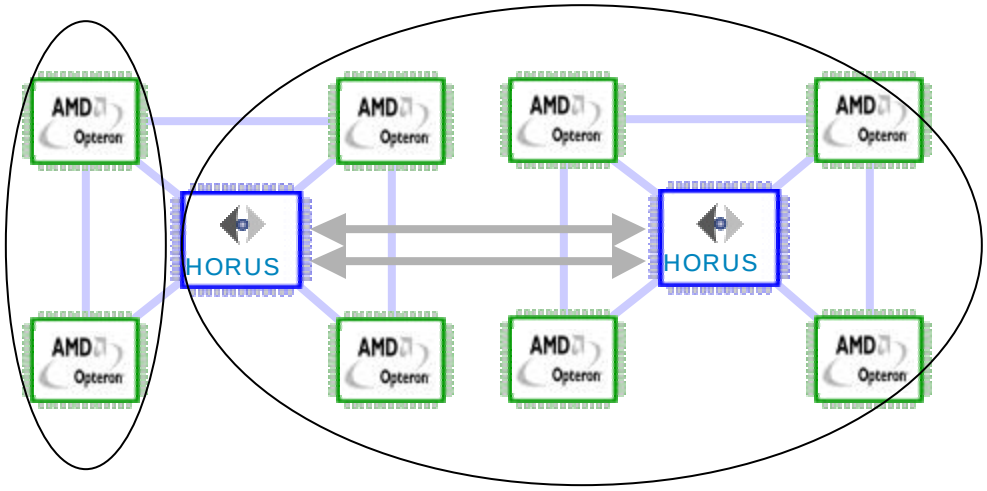
HORUS implements over 50 individual performance counters that can simultaneously monitor transaction flow, cache performance, etc. These performance counters are highly configurable and most counters can be coupled with an address range. These performance counters are used in analyzing and fine tuning several aspects of its design and operation.

Partitioning

The HORUS chip, together with the Newisys system management solution, provides the necessary features needed to perform partitioning. A large-scale SMP system built using HORUS can be partitioned into multiple smaller SMP systems. The only requirements are that each partition needs to have at least one IO chain that has a south bridge. HORUS can reside only in one partition at a time. Transactions from multiple partitions can't share a HORUS.

HORUS and the Newisys system management solution together provide the capability to dynamically partition a system along any remote links. Dynamic partitioning is not supported across a coherent HT link. Opteron processors need a reset to be able to partition across a coherent HT link. Only static partitions are allowed across coherent HT links. Figure 13 shows a two quad, 8 socket system partitioned into two unequal partitions.

Figure 13: Example of a large SMP system partitioned to two smaller unequal SMP systems



Performance Results

Table 1 shows the latencies for different types of transactions in a system that uses HORUS. These latencies are from the Opteron processors' L2 caches and include latencies inside those processors as well, not just pin-to-pin latencies from HORUS to the Opteron processors. These latencies are for the complete transaction up to the first critical data word.

As Table 1 shows, HORUS has an appreciable latency reduction for transactions that exploit the Directory or RDC. Together, RDC and Directory significantly reduce the average transaction latency and also significantly reduce the remote link usage.

Table 1: Latencies in a HORUS-based system with different configurations for different CPU-to-memory-controller requests

Transaction type	Latencies (in ns) in two-, four- and eight-quad systems				Outcome and subsequent action
	2	3	4	8	
L2SM*	269	269	269	334	Directory hit: Probe all remote quads.
L2LM	293	293	293	357	Directory hit: Probe all remote quads.
L2RM	356	396	396	461	Remote data cache (RDC) miss and Directory hit in home quad: Probe all remote quads.
L2SM	96	96	96	96	Directory miss: No probes to remote quad.
L2LM	122	122	122	122	Directory miss: No probes to remote quad.
L2RM	139	139	139	139	RDC hit: Transaction completes locally.

*L2SM represents CPU requests to a memory controller in the same Opteron; L2LM, CPU requests to memory controller in a different Opteron; and L2RM, CPU requests to memory controller in a remote quad.

Figure 14 shows the performance scaling of HORUS for an OLTP application. We used a cycle-accurate performance model and a short sample of traces for TPC-C benchmarks (version 5.1) at steady state. We assumed that Opteron processors were running at 2.8 GHz and had a 400-MHz dual-data-rate (DDR) DRAM and a 1GHz HT link. The HORUS core ran at 500 MHz, with off-chip RDC memory implemented using a 250-MHz DDR-FCRAM. The estimated hit rate for RDC and Directory was 90 percent.

Because of its multiple interfaces, multiple PEs, non-blocking crossbars, dual ports to off-chip memory, 32-entry-deep pending buffers in each PE, and large number of virtual channel credits in each link, HORUS can handle a large number of transactions simultaneously. As the figure shows, increasing the number of outstanding transactions and CPU cores in the Opteron processors decreases the latency added because of HORUS. Thus, the scaling of HORUS improves significantly. Our other performance models support this.

Figure 14 shows the HORUS A.0 bring-up board estimated performance projections for an Opteron-only (glueless dual-core and uni-core) versus Opteron plus HORUS system (HORUS dual core and uni-core) running an OLTP application. (The available Opteron processors depicted are a Newisys estimate, not an AMD commitment.)

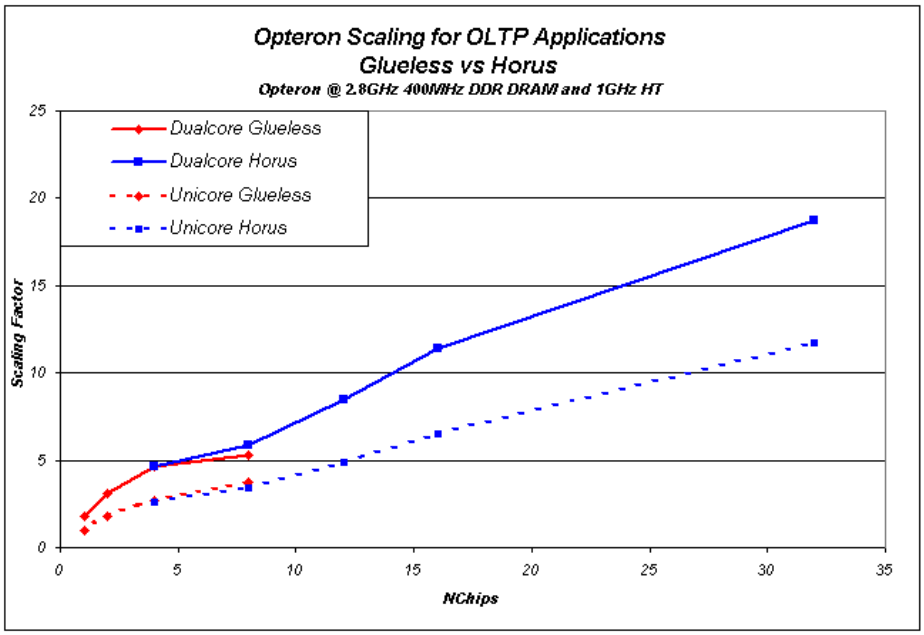
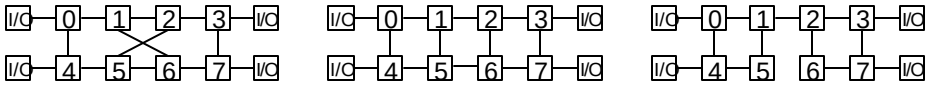


Figure 14: Opteron Scaling for OLTP Applications

Figure 15 shows three possible configurations for an eight-way glueless SMP system. The performance scaling in Figure 14 is with a simple ladder configuration, which relative to the twisted ladder, is easier to build and is modular. On the other hand, the simple ladder has a higher HT link usage than the twisted ladder and so degrades performance.

Figure 15: Configurations for an eight-way glueless SMP system – (a) Twisted ladder (b) simple ladder and (c) dumbbell



(a) (b) (c)

Removing one of the HT links from a simple ladder creates a dumbbell configuration, which scales worse (3.0 for single core and 4.0 for dual core) than a four-way glueless Opteron system because of the excessive traffic on the one HT link connecting the two quads.

Thus, although multiple eight-way configurations are possible, because some links must serve as dedicated I/O, there are fewer links available for constructing the interconnect. Further, routing constraints actually prevent minimum hops in what are otherwise optimal topologies. Our performance studies indicate that an eight-way twisted ladder offers the best trade-off between routing efficiency and I/O bandwidth, although building a twisted-ladder system is expensive, since it requires crossing 4 HT links at the center of the system.

With a twisted-ladder configuration, an eight-way glueless Opteron scales to 4.1 for uni-core Opteron processors and to 6.7 for dual-core versions.

System Management

HORUS provides coherent memory interconnect building blocks. But a complete solution to the single SMP system requires more. Newisys system management provides the following:

- Embedded Service Processor and special interconnect hooks.
- Two Service Processors with independent System Management code (one primary and one redundant in each system).
- System Management code deals with initial setup with include system discovery, configuration determination, configuration input to BIOS, HORUS BIST, memory initialization, loading of the microcode, error handling initialization and system configuration.
- During runtime, system management's responsibilities include reset configuration and distribution, partitioning, various event handling and responses and HORUS debug support.

Summary

Whether it is simple 2-quad scaling or the power of 8-quad scaling – or anything in between – the ExtendiScale architecture using the HORUS ASIC delivers resource flexibility. The ability to adapt on the fly to ever changing compute resource requirements is a powerful weapon for the "do more with less" mantra of IT managers everywhere. The ExtendiScale architecture offers an intuitive design: scalability cables between and amongst nodes.

These evolutionary advances created by the ExtendiScale architecture will enable customers to custom-fit their computing resources to their IT demands in near-real time fashion – without having to purchase this precious headroom in advance. They will enjoy all benefits of horizontal Scale-Out topology – industry standard components, increased availability, near-linear scaling as servers are added – while also being able to shape their flexible computing power into large, single image scaling. Thus, servers based on HORUS can deliver massive memory and I/O scaling to serve hungry database and SMP scaleable applications.